A Categorization of Discovery Technologies for the Internet of Things

Arne Bröring
Siemens AG
Munich, Germany
arne.broering@siemens.com

Soumya Kanti Datta
Communication Systems Department, EURECOM
Sophia Antipolis, France
dattas@eurecom.fr

Christian Bonnet
Communication Systems Department, EURECOM
Sophia Antipolis, France
bonnet@eurecom.fr

ABSTRACT

Discovery of things as well as their resources, metadata, properties, and capabilities is a fundamental requirement in any Internet of Things (IoT) ecosystem. This paper analyzes the state of the art of communication technologies for the IoT with respect to discovery functionalities. Therefore, a comprehensive study of the technology landscape on IoT discovery mechanisms is provided. As a key contribution, we introduce a novel categorization of the available discovery technologies. Further, we identify and analyze the generic interaction pattern of each category. Finally, the technologies of each category are evaluated. With this evaluation at hand, IoT system designers are given decision making support. In the future, this analysis will serve as a basis for a generic discovery framework for the IoT. This work has been elaborated as part of the W3C Web of Things interest group.

CCS Concepts

• Software and its engineering→Software organization and properties • Extra-functional properties→Interoperability • Software system structures→Software Architectures

Keywords

Internet of Things; Discovery; Interaction patterns.

1. INTRODUCTION

The Internet of Things (IoT) is still in its infancy. A lot of data-silos and fragmentations are observed in the IoT market. Recently there is increasing consensus about utilizing the philosophy and standards of the Web to bring harmony in the IoT ecosystems giving rise to Web of Things (WoT). Resource discovery is one of the most fundamental building blocks of a WoT platform. Industry estimates show that 30-50 Billion things will be connected to the Internet by 2020[1] . To provide value-added services to the consumers through a WoT platform, these things must interact with their environment, the cloud and among themselves (paving way for thing-to-thing interactions). The interactions allow things to exchange and process metadata, deriving actionable intelligence, and reacting autonomously to their environment. However, the heterogeneity of things, their capabilities, supported actions, properties and different communication technologies, and protocols add to the complexity of effective realization of the platforms [1].

Therefore, to realize the vision of a truly connected and smart WoT platform, there must be mechanisms available for automatic discovery of resources, their properties and capabilities as well as the means to access them. Depending on the use case, a resource could be a thing, thing metadata or thing description. Furthermore, the discovery mechanisms also depend on other building blocks, such as configuration management, registration and un-registration, or sleep/idle mode of the physical things. The academia, industrial stakeholders, and standard development organizations recognize the importance of discovery and have addressed it from various point of views [2]. For example, discovery can have different scopes based on the contexts of the intended scenarios. When considering from the point of view of physical location and/or network, the discovery scopes can include both local and remote aspects. For example, searching for things around "Me" corresponds to a local scope while a remote scope can be searching for things providing offline maps in a foreign location through a Web service. At the same time, discovery scopes could also be in terms of one-time (thing discovery in a smart home) or long standing (pub-sub style). Similarly, discovery mechanisms also have several dimensions, e.g., context, location, humans (from the access control perspective) etc. An extensive survey of the existing mechanisms and technologies has been carried out to (i) review the current mechanisms used in discovery, (ii) understand different discovery architectures (i.e., centralized directory, peer to peer, or distributed over networks) and (iii) examine the available discovery technologies. Thereby, for the analysis, we focus technologies in this paper, which are from our perspective current practice and most relevant for industry applications.

DOI: http://dx.doi.org/10.1145/2991561.2991570

[1]https://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

The main contributions of this paper are twofold. Firstly, a categorization of current discovery technologies and their interaction patterns is presented in Section 2. Secondly, Section 3 evaluates the identified discovery technologies based on crucial criteria. In Section 4, we show how the categorization and evaluated technologies can be applied in application development. Finally, Section 5 concludes the paper and highlights future works.

2. CATEGORIES OF IOT DISCOVERY TECHNOLOGIES

In the following, we present a novel categorization for IoT discovery technologies. Based on a survey conducted as part of the W3C WoT IG[2], we have identified a set of four technology categories that are related to the discovery of Internet- or Web-enabled things. This set of discovery categories determines a technological landscape for IoT discovery mechanisms. For each of these categories, we name and describe example technologies. As we cannot deliver an exhaustive list of all available technologies, we focus on technologies that are most relevant in today's IoT setups. Also, the descriptions are intentionally kept brief, as they only serve the purpose of explaining why a selected technology is part of this category. More details on these technologies can be gathered by following the provided references.

The foundation of each category is a generic discovery interaction pattern, which we identify and analyze. In each of the discovery interaction patterns we have two central roles: the client and the thing. While the thing represents generally an Internet- or Web-enabled device, the client can e.g. be another thing or a user operated application. The interaction patterns are the basis for a theoretical framework and future abstractions of technology dependencies that are currently hindering interoperable IoT discovery.

2.1 Searching Around Me

This first category includes technologies that enable the discovery of things, which are in close spatial proximity of the client. A common denominator of all comprised technologies is that a spatial "nearness" of the client to the things is required in order to discover the thing. Common to all technologies is the interaction pattern drawn in Fig. 1.

[2] http://www.w3.org/WoT

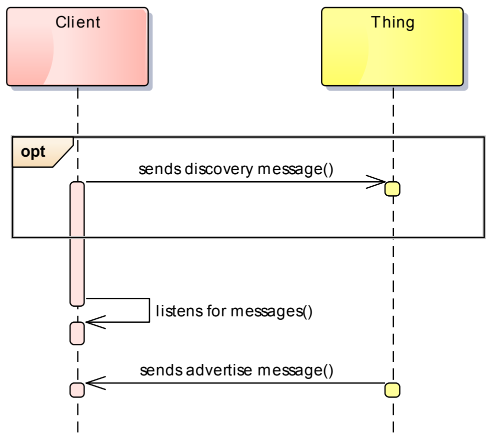

Fig. 1. Interaction pattern: 'Searching around Me'

In the following, we name and briefly describe example discovery technologies which are part of this category.

Near field communication (NFC) [3] technology builds up on RFID by allowing two-way communication between endpoints. It is based on electromagnetic induction. The induction is created between two NFC antennae of the two interacting devices, i.e., client and thing. The range of NFC is very limited (around 10cm) compared to Bluetooth or Wi-Fi. Hence, the *Searching around Me* is also restricted to a very limited spatial proximity. In practice, it is more often utilized for accessing metadata (see Section 2.4). NFC is often used to exchange information for bootstrapping more powerful communication connections (such as Bluetooth) between two devices. NFC follows the pattern shown in Fig. 1. In case of 'passive' NFC, the client used the indicated option and sends a discovery message, which is a carrier field in case of NFC. The thing receives this and may draw power from this carrier field to send the advertise message as a field modulation. In case of 'active' NFC, the initial option is not used and the thing can directly send an advertise message through field modulation, since it is powered.

Another example of this category is Bluetooth Low Energy (BLE) [4]. Such BLE-enabled devices periodically transmit advertising packets. Depending on the vendor of those devices, the advertising packet comprises different payloads. Examples are Google's Eddystone beacon[3], which transmits a Uniform Resource Identifier (URI), or Apple's iBeacon[4], whose advertising packets comprise of a Universally Unique Identifier (UUID). While the iBeacon requires a smartphone app that is enabled to read whitelisted objects and a server to resolve the UUID, the Eddystone beacons can rely on standard Web mechanisms, i.e., a Domain Name System (DNS) server, to resolve the transmitted URI. The interaction style is in case of BLE beacons the push principle, i.e., a beacon broadcasts advertise message, while the client is listening for incoming message. Thus, it follows the pattern shown in Fig. 1.

Another form of searching things around me is markerless recognition. It is an augmented reality (AR) technique using sensors such as accelerometer, camera, compass, and GPS to determine the client's location and field of view. Based on the

[3] https://developers.google.com/beacons/
[4] https://developer.apple.com/ibeacon/

derived position, surrounding things can be discovered. Although useful and a valid method, we do not consider markerless recognition as a discovery technology here, but an application that utilizes the *Searching in Directories* pattern (Section 2.3) for its implementation, as the key discovery mechanism is implemented on the directory part. Specific queries (e.g., client location and field of view) are sent to the directory as part of the query. It is e.g. implemented by AR browsers such as Layar[5] or Aurasma[6].

2.2 Searching on My Network

This category includes technologies that allow discovery of endpoints of things on the network. The according pattern is described in Fig. 2.

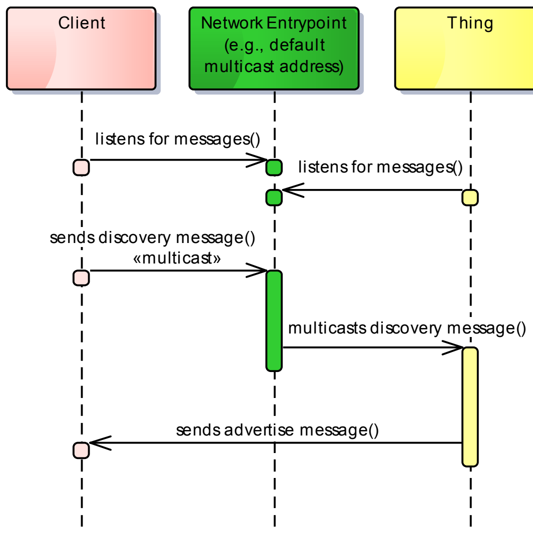
Client
Network Entrypoint (e.g., default multicast address)
Thing
listens for messages()
listens for messages()
sends discovery message() «multicast»
multicasts discovery message()
sends advertise message()

Fig. 2. Interaction pattern: 'Searching on My Network'

Multicast DNS (mDNS) [5] is based on the Internet Protocol (IP) and the User Datagram Protocol (UDP). In a given network, mDNS can be utilized for resolving host names to IP addresses. The interaction model of mDNS conforms to the pattern shown in Fig. 2: In case an mDNS client wants to discover a thing's endpoint by resolving its host name to an IP address, it has to send an IP multicast query message over the network. The message calls the host with that name to reply and identify. Once the host receives the message, it replies via multicast message that contains its IP address. All nodes in the network receiving that multicast message update their mDNS caches accordingly. The mDNS protocol is often used in conjunction with DNS-Based Service Discovery (DNS-SD) [6]. Through this technology, a client can do simple service discovery based on standard DNS queries. mDNS is for example part of the 'zeroconf' technology suite and implemented by Apple's Bonjour[7].

Multicast CoAP is based on IP and UDP. CoAP [7] supports requests to an IP multicast group. Clients can use multicast CoAP and the "all-CoAP-nodes" multicast address to discover things, as CoAP servers. CoAP servers listen on the "all-CoAP-nodes" address and advertise themselves by replying to the client request. In addition, this approach can be combined with the CoRE Link Format [8]. In this case, a CoAP GET request to the appropriate multicast address is made for the '/.well-known/core' resource to receive a list of all available resources. This interaction model is very similar to the one of mDNS and multicast CoAP also conforms to the *Searching on my Network* pattern as described in Fig. 2.

The Simple Service Discovery Protocol (SSDP) [9] is based on IP, UDP, and SOAP and used by Universal Plug and Play (UPnP) [10] for discovery. In order to discover SSDP services, SSDP client multicasts a discovery request to the SSDP multicast channel and port. SSDP services listen on that channel. If an SSDP service receives a discovery request that matches the service it offers, it responds using a unicast response. This interaction style conforms to the pattern shown in Fig. 2. However, while the here described interaction is initiated by the client, some UPnP devices also push info periodically.

Web Service (WS) Discovery [11] is used by OASIS' Device Profile for Web Services (DPWS) and is based on IP, TCP/UDP, and SOAP. The standard specifies multicast discovery via web service-based communication, i.e., via SOAP messages. It specifically aims at avoiding the need for centralized registries in smaller networks. The standard is e.g. implemented by Microsoft's Web Services on Devices API. The SOAP-based message exchange confirms with the identified pattern (Fig. 2) in the so-called "ad hoc mode", however, the discovery happens in two steps: (1.) finding a service and (2.) resolving a target service's address. In the "managed mode", a discovery proxy is introduced in the interaction and unicast messages are exchanged with it as a directory on network-level (see Section 2.3).

2.3 Searching in Directories

In contrast to the above technologies, a central directory can be used for discovery of things and their resources. Queries can be submitted to the directory to search for things and/or resources.

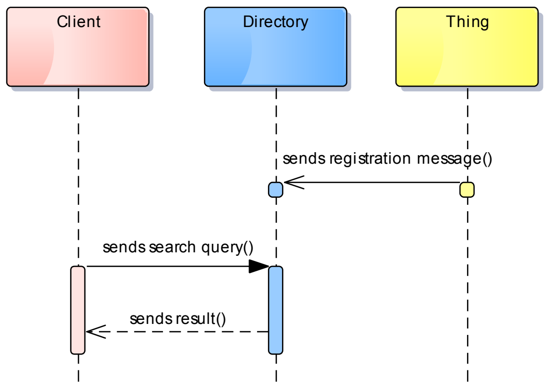

Fig. 3. Interaction pattern: 'Searching in Directories'

The CoRE Resource Directory [12] is described as a draft as part of the CoRE standards family and is based on the CoAP protocol. The Resource Directory (RD) hosts descriptions of CoAP resources held on other CoAP servers, allowing lookups to be performed for those resources. The interaction model of the RD complies with the interaction pattern shown in Fig. 3. The thing registers autonomously (or through third party) its resource

[5] https://www.layar.com/
[6] https://www.aurasma.com/
[7] https://www.apple.com/support/bonjour

descriptions in the CoRE Link Format [8]. Once registered, a thing and its resources can be discovered through a lookup interface that supports simple search queries. For example, a query to search for resources with an associated resource type (“rt”) attribute (e.g., ‘temperature’) can be specified as follows: GET /rd-lookup/res?rt=temperature.

XMPP Internet of Things Discovery [13] is a standards-track document with status ‘experimental’ that describes the behavior and interface of a Thing Registry based on XMPP messages. By sending according XMPP messages, things can be registered at the Thing Registry and it can be searched for things and their metadata (various tags, e.g., location or serial number). A search is performed by providing one or more comparison operators in a search request to the registry. The XMPP Service Discovery [14] standard is utilized as a basis to support XMPP discovery. It allows discovering the identity and capabilities of an entity and associated items. The Thing Registry conforms to the interaction pattern of the Directory shown in Fig. 3.

The HyperCat specification [15] defines a lightweight catalogue that exposes resources encoded in JSON via HTTP on the Web. It lists a number of resource items identified via URI and each described with a number of triple statements, according to the model of the Resource Description Framework (RDF) [16]. The “simple search” extension of the HyperCat catalogue follows the design of a *Directory*, as described in Fig. 3. A central entry point allows to insert new descriptions of thing resources and to submit search queries to perform discovery.

The Sensor Instance Registry (SIR) [17] is a discussion paper at the Open Geospatial Consortium (OGC) and part of the Sensor Web Enablement [18] family of standards. It conforms to the pattern of a *Directory* shown in Fig. 3, by defining a web service interface for registering and searching of metadata and status information of sensing devices (i.e., things). It is capable of automatically harvesting such metadata from defined sources, and transforming the collected metadata sets into a homogeneous data model.

SPARQL is a powerful query language standardized as a W3C recommendation [19]. It can be used to query and manipulate RDF [16] graph content that is stored in an RDF triple store. A SPARQL endpoint of an RDF triple store accepts advanced queries that generally consist of SELECT/WHERE statements for RDF data. In response to a query, the endpoint can return the result in JSON, CSV or XML format. In its basic form, a SPARQL endpoint complies with a *Directory* (see Fig. 3). However, it goes beyond this simple interaction model by supporting for example federated queries

2.4 Accessing Thing Metadata

Once a thing has been discovered with the approaches above, its metadata (e.g., a description and/or measurement metadata) needs to be accessed. That is performed through this category of technologies. Accessing thing metadata can further allow clients to implement a ranking and/or filtering system for the discovered things.

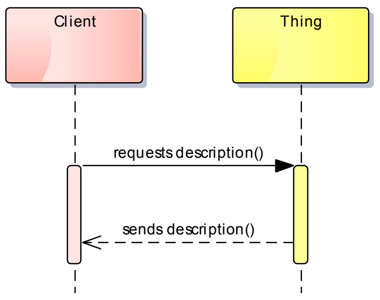

Fig. 4. Interaction pattern: ‘Accessing Thing Metadata’

The CoRE Link Format [8] is an IETF standard that has already been widely explored in research (see e.g., [20], [21]) and has more and more relevance in practice. It is used to access and communicate thing descriptions. Therefore, it provides a mechanism to list URI links to resources such as the thing metadata hosted by a server (e.g., a Web-enabled thing). Attributes can be attached to those resource links to further describe the linked resource. The communication to enable the metadata access through this technology is based on UDP and CoAP: The CoRE Link Format standard defines a default URI ("/.well-known/core") that is offered by the CoAP server (i.e., the thing) and where the list of resource links can be requested. This interaction model complies with the pattern in Fig. 4. The Link Format can also be used to list the items of a central repository as it is done by the CoRE Resource Directory (see Section 2.3).

The Open Geospatial Consortium’s Sensor Observation Service (OGC SOS) [22] is a web service interface that allows to query metadata of sensing devices (i.e. things) as well as their measured data. The OGC SOS standard is part of the Sensor Web Enablement framework [18]. In order to allow accessing the correct metadata description of a thing, OGC SOS also supports advanced temporal filtering. So, this standard complies with the pattern (Fig. 4) and goes beyond with this temporal filtering functionality.

Another way of accessing metadata or a description of a thing is an optical marker tagged onto such a thing. These include visual markers such as barcodes or QR codes, which can be scanned from a client application and then decoded to receive the necessary information. In that sense, the interaction pattern complies with the one in Fig. 4. However, the client needs to be able to scan the optical marker tag on the thing. Since a spatial proximity is required, the interaction is also similar to the *Searching things around Me* pattern presented in Fig. 1. However, optical markers are in general different from technologies such as NFC or BLE, as they only allow to receive more information about a tagged thing, but not to discover a thing. In case of technologies such as NFC and BLE, the device to be discovered may not be directly visible or in reach of the client. Instead, the client “searches” its surroundings for the devices. In case of a QR code, a direct access is possible.

2.5 Research on IoT Discovery Mechanisms

Beyond the above identified discovery technologies, which are already established in industry, there have been various approaches researched in recent years. In the following, we present selected works on advanced IoT discovery solutions.

However, since we focus in this paper on applicable technology, we do not include these in our analysis and evaluation.

One key research direction has been thing discovery through peer-to-peer (P2P) techniques.

In such P2P discovery solutions, the searchable directory is essentially distributed across the peers, which can be e.g. network gateways or the things directly. Typically, this is based on distributed hash tables (DHT). A DHT maps the search space to a numeric range and then allocates servers to parts of that range. The technique works well for scale free networks. It requires peers in the P2P overlay to host parts of the searchable directory, to have full connectivity and computing power to be able to forward overlay messages, and to keep a consistent DHT. P2P overlays tolerate certain amounts of churn; however, it is typically impractical for constrained devices to participate as full peers on the DHT [2].

A recent IETF standard proposes such a mechanism: the IETF RFC 7650 [23] describes CoAP usage for REsource LOcation And Discovery (RELOAD), which is a distributed hash table based P2P protocol. This allows CoAP nodes to store resources in a RELOAD P2P overlay, provides a lookup service. This also enables the use of the RELOAD overlay as a cache for the thing metadata. In contrast to the IETF RFC 7650, the IETF draft on the Distributed Resource Directory (DRD) [24] proposes to utilize raw DHT with no RELAOD dependencies. The distribution is achieved by using a structured P2P overlay.

Cirani et al. [25] describe a scalable and self-configurable architecture for service discovery in IoT. M2M gateways form the backbone of the architecture and allow registration and un-registration of things. The list of registered things is maintained in a CoAP server. In the distributed architecture several gateways are interlinked through two P2P overlays namely distributed local service (DLS) and distributed geographic table (DGT) to facilitate global thing discovery.

Another key research direction on IoT discovery relates to the combination with Semantic Web methodologies.

For example, Zhou and Ma [26] present an ontology for vehicular sensors. Their algorithm calculates semantic similarity as well as relativity and combines them to work out the maximum value of the required concepts of a Web service. Then a matching degree is worked out to discover the relevant Web service.

In [27], the authors introduce a framework based on Semantic Web technology that utilizes the concept of a smart object (i.e. a thing) and their service advertisement that is composed of service metadata including name, id, endpoint, location and semantic annotation link. The authors argue that the advertisement based mechanism facilitates object registration and simplifies discovery.

Another Semantic Web based service discovery methodology is introduced in [28]. The authors propose a middleware which performs discovery using ontologies and deriving contextual information that is inferred from sensor data using reasoning technology.

Malewski et al. [29] propose a Semantic Web approach based on ontology matchmaking to discover required things for plug and play purposes. Going beyond discovery, the approach includes a mediation of advertised things to required things, by using SWRL rules. Accompanying discovery by such mediation ensures that more potentially relevant matches are discovered and can be used.

3. EVALUATION OF IOT DISCOVERY TECHNOLOGIES

In this section, the above mentioned discovery technologies are evaluated against crucial criteria. Those criteria are detailed in the following. Table I below presents the results of the evaluation of the technologies.

- **Bootstrapping** – This criterion defines whether and how a technology provides the means to start interacting with things and their resources after discovery.
- **Range** – This criterion is used for the spatial extent within which a discovery technology is functioning. Generally, this can be either a “local” or a “remote” discovery. A local discovery is e.g. done in case of NFC with a range under 10 cm. A remote discovery is e.g. done in case of SPARQL endpoints where the limiting range is the scope of the directory.
- **Basic Search** – This criterion defines if a technology allows submitting search queries based on terms of the underlying information model. This includes searches such as e.g.: 'search all things with name 123 AND with property temperature'.
- **Richness of Queries** - This criterion specifies to what extent contextual query parameters can be passed in a search query. E.g., this can include spatial parameters ('search all things in NYC') or temporal parameters ('search all things active yesterday'). This goes beyond basic search queries that allow searching based on terms of the underlying information model. Since there is a large spectrum of search query designs, we distinguish between 4 options in this criterion: “no”, “limited”, “medium”, and “high”. Thereby, “limited” means that the search query has a tagging mechanism. The “medium” option characterizes search query designs that offer search operators (e.g., spatial, temporal, or aggregate functions). Search interfaces can be characterized with “high” in richness if they allow a very flexible assembly of queries (such as logical concatenation or advanced functions).
- **Ranking of Results** - This criterion specifies how/if the discovery mechanism is capable of ranking search results.

4. APPLYING THE IOT DISCOVERY CATEGORIZATION

This section outlines how the categorization and evaluation of IoT discovery technologies presented above can be utilized in IoT application development.

To illustrate this, an example application scenario is assumed: A company plans to organize a marathon and needs a system that enables discovery of the runners and their virtual counterparts (as *things*) as well as their wearables devices. The runners should be discovered locally at checkpoints. Through the wearable devices, vital body parameters (e.g., heart rate, blood pressure, hydration level) can be collected and monitored. The monitoring system must be able to rank the results of discovery based on if a runner needs medical attention. To accomplish that, once the runner is identified locally, the discovery system should be able to answer rich discovery queries (e.g., about her/his performance over time, or vital parameters).

To approach this application, the identified categories have been placed in a multi-entry decision tree (see Fig. 5). First, appropriate local discovery mechanisms are identified. Thereby, the particular local range of the technology is the key determinator. Hence, the entry point of the decision tree is "Range". Since 10 cm is too little range, and vicinity to the runner (e.g., via optical marker on jersey) cannot be guaranteed, the decision falls on "Local within 100m". This means, runners can be equipped with wearable devices using, e.g., BLE chips. Second, a directory is needed to store the registration information of runners and their wearable devices. As mentioned earlier, the system also needs to support richness in queries. For example, an administrator overseeing the marathon might be able to search for runners with "very high blood pressure". Also ranking of results is important to identify who needs medical attention right away and what is his/her current location.

Therefore in this application scenario, the runners need wearable devices equipped with BLE; however, that alone is not sufficient. The discovery mechanism also needs to support remote discovery through a directory, a "high" richness of queries, and needs to allow ranking of results. Looking at the decision tree in Fig. 5 (based on input from Table 1) the discovery technology to be selected for the directory is a SPARQL endpoint, since it is the only technology that supports "high" richness of queries and ranking of results.

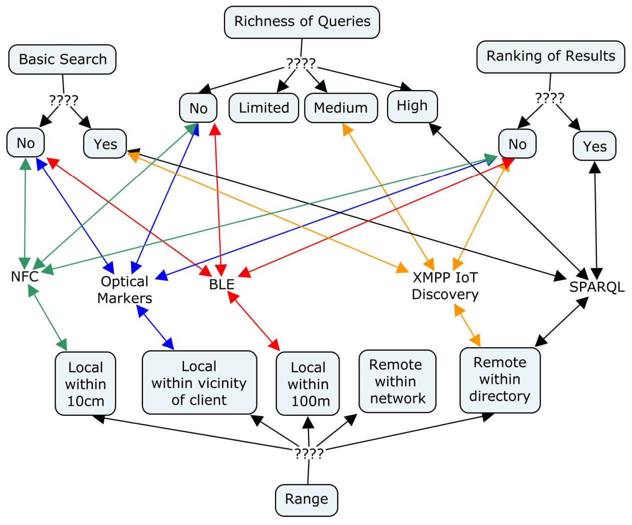

Fig. 5. Multi-entry decision tree for IoT discovery technology selection (Note: for better brevity only a small subset of technologies has been incorporated in the tree).

5. CONCLUSIONS AND OUTLOOK

This paper analyzes discovery technologies as a key part of IoT ecosystems. Looking at current practices and relevance for industry, we have identified available IoT discovery technologies. Further, we grouped those technologies into four key categories. Each category has been well explained and example technologies belonging to each category have been named. As a key contribution of this paper, we have identified a generic interaction pattern for each of the four categories. Building up on this categorization, we have evaluated the discovery technologies with respect to five crucial capabilities for discovery: bootstrapping, range of discovery, search and richness of queries, as well as ranking of results. This evaluation has shown great differences in the analyzed technologies. This shows their individual applicability in different applications.

We have shown how this categorization can already be used by application developers in decision making which discovery technology to be used for given requirements. In the future, the application of this analysis can go beyond the development time as shown in the application here. This categorization may serve as a basis for *generic discovery frameworks* for the IoT, which support a broad range of IoT discovery technologies. Using such a framework, the concrete discovery technology could be selected by an application at runtime. A generic IoT discovery framework can abstract from the concrete underlying discovery technologies and would make it opaque to the developer which technology to be used. This would facilitate IoT application development and would decrease standard dependency.

While the above described prospects show great value, one needs to consider security aspects. Standardized discovery frameworks will facilitate application development. Nonetheless, there is also higher a risk of security threads, once mechanisms are established and widely used. Hence, future research needs to also tackle those challenges and find solutions towards improving security in the various discovery technologies.

ACKNOWLEDGMENTS

This work is an outcome of the W3C Web of Things Interest Group[8]. We are very thankful for many fruitful discussions with the IG's members. The work has been partly funded by the ITEA3 project "Building as a Service" (BaaS)[9] funded by the German Federal Ministry of Education and Research (BMBF) under reference number 01IS13019A, and the French research project DataTweet (ANR-13-INFR-0008)[10].

6. REFERENCES

[1] A. J. Jara, P. Lopez, D. Fernandez, J. F. Castillo, M. A. Zamora, and A. F. Skarmeta, "Mobile digcovery: A global service discovery for the Internet of Things," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, 2013, pp. 1325–1330.

[2] S. K. Datta, R. P. F. Da Costa, and C. Bonnet, "Resource Discovery in Internet of Things: Current Trends and Future Standardization Aspects," in *IEEE 2nd World Forum on Internet of Things (WF-IoT), 2015*, Milan, Italy, 2015, pp. 542–547.

[3] R. Want, "Near field communication," *IEEE Pervasive Comput.*, no. 3, pp. 4–7, 2011.

[4] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11734–11753, 2012.

[8] http://www.w3.org/WoT
[9] http://baas-itea2.eu
[10] http://www.agence-nationale-recherche.fr/?Projet=ANR-13-INFR-0008

[5] S. Cheshire and M. Krochmal, *IETF RFC 6762: Multicast DNS*. IETF, 2013.

[6] S. Cheshire and M. Krochmal, *IETF RFC 6763: DNS-based Service Discovery*. IETF, 2013.

[7] C. Bormann, A. P. Castellani, and Z. Shelby, “CoAP: An Application Protocol for Billions of Tiny Internet Nodes,” *Internet Comput.*, vol. 16, no. 2, pp. 62–67, 2012.

[8] Z. Shelby, *IETF RFC 6690: Constrained RESTful Environments (CoRE) Link Format*. IETF, 2012.

[9] Y. Goland, T. Cai, P. Leach, and Y. Gu, *IETF Internet Draft: Simple Service Discovery Protocol/1.0 Operating without an Arbiter*. IETF, 1999.

[10] M. Jeronimo and J. Weast, *UPnP design by example: a software developer’s guide to universal plug and play*. Intel Press, 2003.

[11] T. Nixon, A. Regnier, V. Modi, and D. Kemp, *OASIS Web Services Dynamic Discovery (WS-Discovery) Version 1.1*. OASIS, 2009.

[12] Z. Shelby, M. Koster, C. Bormann, and P. van der Stok, *IETF Internet Draft: CoRE Resource Directory*. IETF, 2016.

[13] P. Waher and R. Klauck, *XEP-0347: Internet of Things - Discovery*. XMPP Standards Foundation, 2015.

[14] J. Hildebrand, P. Millard, R. Eatmon, and P. Saint-Andre, *XEP-0030: Service Discovery*. XMPP Standards Foundation, 2008.

[15] T. Jaffey, J. Davies, and P. Beart, *Hypercat 3.00 Specification*. Hypercat Limited, 2016.

[16] R. Cyganiak, D. Wood, and M. Lanthaler, *RDF 1.1 Concepts and Abstract Syntax*. W3C, 2014.

[17] S. Jirka and D. Nüst, *OGC Discussion Paper 10-171: Sensor Instance Registry*. Wayland, MA, USA: Open Geospatial Consortium, 2010.

[18] A. Bröring, J. Echterhoff, S. Jirka, I. Simonis, T. Everding, C. Stasch, S. Liang, and Rob Lemmens, “New Generation Sensor Web Enablement,” *Sensors*, vol. 11, no. 3, pp. 2652–2699, 2011.

[19] S. Harris, A. Seaborne, and E. Prud’hommeaux, *SPARQL 1.1 Query Language*. W3C, 2013.

[20] S. K. Datta and C. Bonnet, “Smart M2M gateway based architecture for M2M device and Endpoint management,” in *Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing (CPSCom), IEEE*, 2014, pp. 61–68.

[21] S. K. Datta and C. Bonnet, “A lightweight framework for efficient M2M device management in oneM2M architecture,” in *Recent Advances in Internet of Things (RIoT), 2015 International Conference on*, 2015, pp. 1–6.

[22] A. Bröring, C. Stasch, and J. Echterhoff, *OGC Implementation Standard 12-006: OGC Sensor Observation Service Interface Standard, Version 2.0*. Wayland, MA, USA: Open Geospatial Consortium, 2012.

[23] J. Jimenez, J. Lopez-Vega, J. Maenpaa, and G. Camarillo, *IETF RFC 7650: A Constrained Application Protocol (CoAP) Usage for REsource LOcation And Discovery (RELOAD)*. IETF, 2015.

[24] J. Jimenez, M. Liu, and E. Harjula, *IETF Internet Draft: A Distributed Resource Directory (DRD)*. IETF, 2013.

[25] S. Cirani, L. Davoli, G. Ferrari, R. Léone, P. Medagliani, M. Picone, and L. Veltri, “A scalable and self-configuring architecture for service discovery in the internet of things,” *Internet Things J. IEEE*, vol. 1, no. 5, pp. 508–521, 2014.

[26] M. Zhou and Y. Ma, “A web service discovery computational method for IOT system,” in *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, 2012, vol. 3, pp. 1009–1012.

[27] S. Alam and J. Noll, “A semantic enhanced service proxy framework for internet of things,” in *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int’l Conference on & Int’l Conference on Cyber, Physical and Social Computing (CPSCom)*, 2010, pp. 488–495.

[28] C. H. Yun, Y. W. Lee, and H. S. Jung, “An evaluation of semantic service discovery of a U-city middleware,” in *Advanced Communication Technology (ICACT), 2010 The 12th International Conference on*, 2010, vol. 1, pp. 600–603.

[29] C. Malewski, A. Bröring, P. Maué, and K. Janowicz, “Semantic matchmaking & mediation for sensors on the sensor web,” *Sel. Top. Appl. Earth Obs. Remote Sens. IEEE J. Of*, vol. 7, no. 3, pp. 929–934, 2014.

Table I: Evaluation of IoT Discovery Technologies

Category	Technology Name	Bootstrapping	Range	Basic Search	Richness of Queries	Ranking of Results
Searching things around me	NFC	Yes, pointing to thing endpoint.	Local within 10 cm	No.	No.	No.
	BLE	Yes, pointing to thing endpoint.	Local within 100 m	No.	No.	No.
Searching things on my network	mDNS	Yes, client receives thing's IP address.	Remote within network	No.	No.	No.
	Multicast CoAP	Yes, client receives thing's resource description.	Remote within network	No.	No.	No.
	SSDP	Yes, client receives endpoint of UPnP device.	Remote within network	Yes, but only basic filtering for devices of specific type.	No.	No.
	WS Discovery	Yes, two steps: (1.) finding services and (2.) resolving their address.	Remote within network	Yes, basic querying (e.g., on service type) is possible.	Limited, querying using "scopes" is possible.	No.
Searching in directories	CoRE Resoure Directory	Yes, client receives thing's resource description.	Remote within directory	Yes, key-value-pair search based on tagged resources.	Limited, simple tag concatenation.	No.
	XMPP IoT Discovery	Yes, provides various metadata for discovered thing.	Remote within directory	Yes, as part of search query message.	Medium, several search operators are defined.	No.
	HyperCat	Yes, provides reference to thing.	Remote within directory	Yes, search based on given tags.	Limited, key-value-pairs can be defined as query.	No.
	Sensor Instance Registry	Yes, provides rich metadata description (SensorML).	Remote within directory	Yes, bound to SensorML as metadata model.	Medium, allows spatial and temporal queries.	No.
	SPARQL Endpoint	Yes, can potentially provide rich description of discovered thing.	Remote within directory	Yes, flexible search interface.	High, very flexible design of queries.	Yes.
Accessing thing metadata	CoRE Link Format	Yes, client receives links to thing's resources.	Remote within directory	No.	No.	No.
	OGC SOS	Yes, it returns SensorML when metadata is queried.	Remote within directory	No.	Limited, allows temporal queries to access metadata.	No.
	Optical Markers	Yes, pointing to thing endpoint.	Local within vicinity of client.	No.	No.	No.